MERIDIAN
SOFTWARE SYSTEMS

Getting Started
PC DOS

Meridian Licensing, Registration and Support

Software License Agreement

The enclosed computer software, the media on which it is provided, and any associated documentation ("Software") is licensed by Meridian Software Systems, Inc. ("Meridian") to the original customer ("User") for use by the original customer only in accordance with the terms set forth below.

1. LICENSE: In exchange for a License fee, Meridian hereby grants to User a personal non–exclusive and non–transferrable license to use the Software and associated documentation subject to the terms and conditions set forth in this License Agreement.

2. RESTRICTIONS ON USE AND TRANSFER: The original and any backup copies of the Software are to be used only in connection with a single computer. You may physically transfer the Software from one computer to another, provided that the Software is used with only one computer at a time. User may make no more than three (3) backup copies solely for use as backup and not to be sold, loaned, leased, gifted or otherwise transferred or used by any other person.

User may not (a) make or distribute copies or translations of the Software or documentation to others, except that you may install the Software on a hard disk and make the three backup copies allowed under this Agreement; (b) rent, lease, or grant sublicenses or other rights for the Software; (c) provide use of the Software in a computer service business, network, timesharing, interactive cable television, multiple CPU or multiple user arrangement to users who are not individually licensed by Meridian; (d) make telecommunications data transmissions of the Software; (e) disassemble, reverse–engineer, or attempt to disassemble or reverse–engineer any part of the Software; (f) remove any notice of proprietary rights (i.e. copyright, patent, trademark, trade name, etc.) from the Software or packaging therefor.

3. Government and Government Contractor Licenses: If User is a unit or agency of the United States Government, or a contractor that will or may supply the Software to any unit or agency of the United States Government, User agrees that: (a) the Software is "Commercial Computer Software" as that term is defined in Paragraph 27.401 of the DoD Supplement to the Federal Acquisition Regulations and in any corresponding DAR and ASPR Regulations, or any subsequent laws or regulations or revisions thereto; (b) Meridian represents to User that the Software is developed at private expense, and no part of it was developed with government funds; (c) the government's use of the Software is subject to "Restricted Rights" as that term is defined in paragraph 52.227–7013 of the DoD Supplement or any subsequent laws or regulations or revisions thereto; (d) Meridian represents that the Software may be deemed a trade secret of Meridian for all purposes including for the purpose of the Freedom of Information Act; (e) each copy of the Software supplied to the government or to another government contractor will also bear and/or have affixed near it the restricted rights legends as supplied by Meridian; (f) User will indemnify Meridian and hold it harmless against any liability, loss, costs, and expenses (including reasonable attorney's fees) arising out of any breach by User of the provisions of this License Agreement.

4. Disclaimer of Warranties: THE SOFTWARE PROVIDED HEREUNDER IS LICENSED ON AN "AS IS" BASIS. MERIDIAN SOFTWARE SYSTEMS, INC. SPECIFICALLY DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. NO ORAL OR WRITTEN STATEMENTS, REPRESENTATIONS OR OTHER AFFIRMATION OF FACT, INCLUDING BUT NOT LIMITED TO, STATEMENTS REGARDING CAPABILITY, CAPACITY, SUITABILITY FOR USE OR PERFORMANCE OF SOFTWARE SHALL BE RELIED UPON BY USER OR DEEMED TO BE A WARRANTY OR REPRESENTATION BY MERIDIAN FOR ANY PURPOSE, OR GIVE RISE TO ANY

LIABILITY OR OBLIGATION OF MERIDIAN WHATSOEVER. THE USER ACCEPTS ALL RESPONSIBILITY FOR SELECTING THE SOFTWARE TO MEET USER NEEDS OR SPECIFIC PURPOSES.

5. Agreed Limitation of Liability: If, at the time of delivery, there are any defects in the media on which the Software is provided, User's sole and exclusive remedy shall be the replacement of any media returned to Meridian within ninety (90) days of receipt of the Software by User, or, at Meridian's sole election and expense, a refund of the License fee paid to Meridian. IN NO EVENT SHALL MERIDIAN BE LIABLE FOR ANY LOSS OF PROFIT OR ANY OTHER COMMERCIAL DAMAGE, INCLUDING BUT NOT LIMITED TO INDIRECT, SPECIAL, INCIDENTAL, CONSEQUENTIAL OR OTHER DAMAGES (INCLUDING BUT NOT LIMITED TO LOST PROFITS), EVEN IF MERIDIAN HAS NOTICE OF THE POSSIBILITY OF SUCH DAMAGES. MERIDIAN'S LIABILITY, IF ANY, FOR DAMAGES ARISING OUT OF BREACH OF CONTRACT, TORT, PATENT OR COPYRIGHT INFRINGEMENT, OR ANY OTHER CAUSE OF ACTION, SHALL NOT EXCEED THE LICENSE FEE PAID BY THE USER FOR THE SOFTWARE HEREUNDER, NOTWITHSTANDING ANY FINDING THAT THE EXCLUSIVE REMEDY REFERRED TO ABOVE FAILED OF ITS ESSENTIAL PURPOSE.

No action, regardless of form, arising out of any claimed breach of this License or relating to transactions which are the subject of the License may be brought by either party more than one (1) year after the delivery of the Software.

6. Term: Upon receipt of the License fee, this License becomes effective and remains in full force and effect for twenty (20) years or until terminated. The License may be terminated at any time by destroying the Software and documentation and all copies in whatever form. It will also terminate if User fails to comply with any term or condition of this License Agreement. User agrees upon such termination to destroy the Software and documentation together with all copies in whatever form.

7. Governing Law: This License Agreement shall be construed, interpreted, and governed by the laws of the State of California. If any portion of this Agreement is held to be invalid by a court of competent jurisdiction, the validity of the remaining provisions shall not be affected thereby.

8. Entire Agreement: This License Agreement constitutes the entire agreement between Meridian and User with respect to the Software and supersedes all prior communications with the User, including all oral or written proposals. This License may not be modified except in writing by a duly authorized representative of Meridian.

Table of Contents

Contents

Chapter 1 Before You Begin

The software contains the following components:

- ***Meridian Ada Compiler Environment (ACE)***
  The Meridian Ada Compiler Environment (ACE) is a menu driven, multiwindow interface to the Meridian Ada Compilation System. ACE has a powerful text editor, extensive hypertext online help for the editor and compiler, an online Reference Manual for the Ada Programming Language (LRM), Ada Syntax driven editing capability, and ability to customize features to interface to your own tools.

- ***Meridian Ada Compiler***
  The Meridian Ada compiler is a production quality Ada compiler validated by the AJPO in accordance with its current testing procedures. The compiler includes complete support for the Language Reference Manual Chapter 13 facilities as well as all standard support packages. Pragma interface for C and assembly languages is also supported. The support for tasking includes pre–emptive scheduling.

- ***The Meridian Ada Debugger***
  The Meridian Ada Debugger is an interactive source–level debugger for use with programs written using the Meridian Ada compiler. The Debugger allows you to debug in high–level Ada terms; no knowledge of the underlying machine architecture is required.

- ***The Meridian Ada Utility Library***
  The Meridian Ada Utility Library is an additional set of Ada Packages defining facilities for accessing command line arguments, doing efficient bit manipulations, performing transcendental math functions, and manipulating variable length strings and large arrays.

- ***The DOS Environment Library***
  The DOS Environment Library gives programs full access to the MS–DOS system facilities which are not otherwise readily available to Ada programmers. These facilities include screen management and other BIOS functions.

- ***Program Optimizer***
  The optimizer analyzes the compiler's internal representation of a user's program and performs a number of code–improving transformations ranging from local optimizations to global subprogram removal that result in enhanced output code.

Important: The *Meridian Ada Compiler User's Guide* describes the full compilation system. Items identified as Groupware or Extended Mode Features are not included in the software.

1.1 Package Contents

The disks contain the following software:

- ACE
- The Meridian Ada compiler which includes the Debugger, Optimizer, and Ada Utility Library
- DOS Environment Library

- Math Library source
- Graphics Library source
- Booch Component Sampler

1.2 Inside This Manual

Here's what you'll find inside this manual:

- **Chapter 1, "Before You Begin"** provides general information about the software and some background information about the product. This chapter recommends a path that gets you started as quickly as possible.
- **Chapter 2, "Customer Service"** provides information about our support programs as well as information on utilizing technical support.
- **Chapter 3, "Installing the software"** tells you how to install the software.
- **Chapter 4 "Troubleshooting"** lists possible solutions to commonly encountered installation problems.

1.3 Documentation Overview

A list of documents and manuals along with a short overview of each follows:

- *Readme First* provides information you'll need to get started.
- *Getting Started* contains the technical support information along with the installation instructions.
- *Meridian ACE User's Guide* contains the information needed to use ACE. This manual includes tutorials on using ACE.
- *Meridian Ada Compiler User's Guide* contains the information needed to use the compiler. This manual includes the documentation for using the Meridian Ada Utilities Library, standard packages, and the debugger.
- *Reference Manual for the Ada Programming Language ANSI/MIL-STD-1815A-1983* contains information used in programming with Ada.
- *DOS Environment Library User's Guide* contains the information needed to use the interface to the DOS operating system.

1.4 Where To Begin

To Familiarize Yourself with ACE . . .

The Tutorials chapter of the *Meridian ACE User's Guide* contains tutorials to get you started using the editor, compiler and language expansion capability from within ACE.

To Familiarize Yourself with the Compiler . . .

Refer to the *Meridian Ada Compiler User's Guide.*

To Familiarize Yourself with the Debugger . . .

Refer to the *Meridian Ada Compiler User's Guide.*

To Familiarize Yourself with the DOS Environment Library . . .

Refer to the *DOS Environment Library User's Guide.*

1.5 Getting Help

You can display help at any time while using ACE. Just press Alt–H while a menu item is highlighted to display information about that item.

Press the down–arrow key to scroll through the Help text. As you do, topics on the screen for which more information is available are highlighted. Press ENTER to display the information for a highlighted term.

When you are finished, press ESC to "back out" of the sequence of Help screens that have been displayed.

For an in depth discussion of the Help function, see the *Meridian ACE User's Guide*.

1.6 Symbols and Command Format Conventions

The following format conventions are used inside the manual set:

- Command names appear in typewriter bold font.
- Information (prompts and messages) that appears on the screen is shown in typewriter bold font.
- Variable information (information to be entered by you) appears in italics.

See the Tutorials chapter in the *Meridian ACE User's Guide* for format conventions that specifically apply to that chapter.

See the Meridian Ada Command Details chapter in the *Meridian Ada Compiler User's Guide* for Meridian Ada Compiler command specific format conventions.

1.7 Meridian Software Systems

Meridian Software Systems, Incorporated is a company with an established record in high level languages and advanced software development tools. The Meridian Ada compiler is a product of Meridian's years of experience with portable high level language compiler technology. Meridian Ada is U.S. DoD–validated on a variety of different Macintosh, PC DOS, UNIX, and VMS platforms. Contact your sales representative for a complete list of supported hardware.

Chapter 2 Customer Support and Service

2.1 User Registration

To take advantage of Meridian's customer services, you should become a registered user. If you have not already done so, please take a moment now to fill out and return to us the postage–paid REGISTRATION CARD.

Please write your registration number here for future reference.

Registration #:

2.2 Installation Assistance

For ninety days after purchasing a new product, Meridian support personnel provide free installation assistance for those customers who may encounter difficulty with a particular system configuration. All new product sales also include a ninety day warranty on media.

A section which describes common installation problems is included in the Troubleshooting chapter of this manual. Please refer to that chapter if you are having problems installing the software.

2.3 Customer Support

Meridian uses its best efforts coupled with the most advanced software technology available to produce its products, but we recognize that no product is ever completely free of problems and we have established procedures to deal with any problems that may occur. Meridian provides several support programs for the various Meridian products. Support options include:

- Installation Assistance
- Software User Reports
- Phone Support
- Ada Language Consulting
- Bulletin Board with downloadable bug fixes

The available support programs are summarized below. Consult the Meridian Product Support Options product sheet or contact Meridian for detailed prices and specifications.

2.3.1 Premium Customer Assurance Plan

The Premium Customer Assurance Plan (CAP) offers the user an exceptional level of support service for one year, including:

- Updates (minor releases between validations) and Upgrades (new validation releases)

- Unlimited phone support
- "Problems solved" list (describes fixes and workarounds for user–reported problems)
- Meridian Newsletter
- Access to the support bulletin board

All bundled products are supported via CAP services.

CAP services become effective immediately after purchase.

2.4 Contacting Customer Support

The following sections contain some suggestions to make your contact with technical support go smoother. A little preparation before you call will save you and the customer support representative valuable time.

2.4.1 Problem Determination

If you experience difficulty using the software, first try to determine the nature of the problem. Is it language related or compiler related? Ada is a complex programming language with many subtleties. What may appear at first glance to be a compiler problem may in fact be a programming problem. Check your Ada Language Reference Manual to be sure. If a problem turns out to be language related, you need to re–write your program to be legal Ada and push ahead with your development.

If you have determined that a problem appears to be compiler related, try to isolate the cause. What data types are involved? What executable statements are involved? What kinds of program units (procedures, functions, packages, and so on) are involved? Try to create the smallest possible program that illustrates the problem.

Once you have isolated the problem, you can either send a written report to Meridian as described below or phone technical support for a more immediate response.

Once a problem has been isolated, regardless of whether it is language related or compiler related, you should have enough information to attempt to work around the problem. Try alternative implementations of the same idea and continue with your development.

The *Meridian Ada Compiler User's Guide* contains a trouble shooting and helpful hints section that may be of assistance.

2.4.2 Preparing to Call Support

Before you call technical support, you need to be have some specific information available. You might want to make a copy of the software user report form and fill it out before calling as chances are that your technical support representative will need the information requested there. A software user report form is located in this chapter as well as in the back of each manual. An electronic copy is also provided with the software installation diskettes or tape. A list of minimal information follows.

- product registration number
- the software product name and version number
- a concise description of the problem or suggestion
- relevant sample programs

2.4.3 If You Prefer to Contact Us by Mail

If you prefer to contact us by mail, you can use the software User Report Form located later in this chapter or at the back of each user manual.

If you have suggestions for improvements to any of the software products, please notify Meridian in writing. Written reports to Meridian should be sent to the attention of the Technical Support Manager.

Software User Reports will be analyzed by our development people and appropriate action taken. Unless you specify otherwise, we may contact you for more information.

2.5 Our Address

You can reach Meridian by phone, regular mail and electronic mail.

To contact Meridian by regular mail, use the following address:

Meridian Software Systems, Inc.
10 Pasteur St.
Irvine, CA 92718

Meridian's phone number is:

(714) 727 – 0700

Send Fax communications using the following number:

(714) 727 – 3583

Technical Support can be reached at the following number:

(714) 727 – 7070

Meridian can also be reached by electronic mail using one of the following addresses:

- Internet users:

  `support@Meridian.COM`

- If you do not have Internet name servers, you may be able to use the following address:

  `support%Meridian.COM@uunet.uu.net`

- UUCP users (without connections to the Internet) may be able to send mail using the following addresses:

  `geode!support`

  or

  `ucivax!geode!support`

  Ucivax is reachable from the following UUCP nodes:

  `felix laxsqnt sunkist ucbvax ccicpg`

DOS Software User Report Form

Date: _____ Registration #: _____

Name: _____

Company: _____

Address: _____

_____

Telephone: (____)_____

Product: _____ Version #: _____

System Configuration

Machine (XT, AT, PS/2-60, ...): _____

DOS Version #: _____

Base Memory (KB RAM): _____

Extended Memory (MB RAM): _____

80x87 installed: ___ yes ___ no

CPU (8088, 80286, ..): _____

Hard Disk Space (MB): _____

Amount of memory available to application (KB RAM): _____

Problem Reporting

Can the problem be reproduced at will? ___ yes ___ no

- a concise description of the problem
- a small sample program that demonstrates the problem
- any other information that you think might be relevant to the problem

Send completed forms to:

Technical Support
Meridian Software Systems, Inc.
10 Pasteur St.
Irvine, CA 92718

Internet: `support@Meridian.COM`
UUCP: `meridian.com!support`
or `geode!support`

When not using electronic mail, please deliver source code of all sample programs on diskettes.

Thank you for your interest. We appreciate your cooperation in the improvement of our products.

Please duplicate this form for each problem that you are reporting.

Chapter 3 Installing the Software

3.1 Introduction

This chapter describes how to install the the software software on computers that run MS–DOS or PC–DOS.

Warning: Before installing the software, please be sure that you have installed DOS correctly by following the exact installation procedures in the DOS Manuals.

3.2 Quick Start Instructions

If you are an experienced user or have installed a Meridian compiler before, you may be able to install using the simple instructions below.

System Requirements

- MS–DOS or PC–DOS version 2.1 or later.
- 640K bytes of Random Access Memory (RAM) in the base memory area.
- A hard disk with at least 5MB free space.

Installation Instructions

These instruction assume that you are installing the software in `c:\ada`. If you are not installing in `c:\ada`, make the appropriate substitutions in the instructions which follow.

1. Modify the `autoexec.bat` file to add the the software program directory `c:\ada\bin` to the `path` setting.
2. Modify the `autoexec.bat` file to set the DOS environment variable for the ACE directory:

   ```
   set acedir=c:\ada\ace
   ```

3. Run the `install` program using a command with this form:

   ```
   a:install c:\ada
   ```

   This causes the software software to be installed on drive `c:` in the directory `\ada`.

3.3 System Requirements

The system onto which the software is to be installed must possess:

- A processor compatible with the Intel 8086 (e.g. 8088, 8086, 80186, 80286, 80386).
- MS–DOS or PC–DOS version 2.1 or later.
- 640K bytes of Random Access Memory (RAM) in the base memory area.
- When using the compiler, please make certain that you do not have print spoolers, window managers, large device drivers (e.g. network drivers), and other base

memory consuming programs active in your system when you are compiling Ada programs.

- A hard disk (typically 10, 20 or 30MB) with at least 5MB free space.

Meridian Software Systems does not supply MS–DOS, PC–DOS or hardware products. Most 8086, 80286, and 80386–based computers are already initially configured with at least 640K bytes of RAM, but upgrades for smaller systems configured with less memory are easily obtained from computer stores or computer mail order firms.

Complete installation of the compiler system requires approximately 5MB of free space on the disk onto which the system is to be placed. The `dir` command can be used to find the amount of free disk space currently available; it is the last line printed in any directory listing.

3.4 Installation Overview

Installing the Meridian Ada compiler takes several steps:

1. If *Release Notes* accompany the software, read these *Release Notes* for any additional installation information that may apply. The *Release Notes* may provide additional information or replace these installation instructions.
2. Decide where to install the software.
3. Ensure that there is sufficient disk space.
4. Modify the `autoexec.bat` file to add the software program directory to the `path` setting and define the `acedir` environment variable (see section 3.7).
5. Change the `files` option in `config.sys` (see section 3.8).
6. Reboot the system to activate changes (see section 3.9).
7. Load the files onto the hard disk (see section 3.10).
8. Verify the installation (see section 3.11).
9. Test the software (see section 3.12).

These steps are described in order in the sections that follow; a summary (an example only), is given in section 3.13.

There is a brief troubleshooting guide in Chapter 4.

Note: The drive designator `c:` is given only as an *example* in some of the commands that are shown in the succeeding sections. The drive designator may differ, depending on the drive on which the software is installed (e.g. `d:` or `e:` may be used instead).

3.5 Decide Where to Install the Software

Before performing the installation, you must decide where to install the software. The software is to be installed on a particular hard disk (e.g. `c:`) and in a particular directory on that hard disk (e.g. `\ada`).

The full name of the installation directory has the form:

d:\directory

- *d* is the hard disk on which to install the compiler support directories. This should be a letter in the range `b` to `z`. On IBM systems, drive `c` is normally the hard disk drive. On a few others, drive `e` is sometimes the hard disk drive.
- *directory* is the directory in which the software is to be installed. Note that the *directory* must start with a backslash ("\").

An example of this installation directory name is `c:\ada`.

Under the directory in which you decide to install the software, these sub–directories will be created by the installation process:

```
bin
paclib
test
```

If, for example, you decide to install the software in `c:\ada`, then the installation process will automatically create these directories:

```
c:\ada
c:\ada\bin
c:\ada\paclib
c:\ada\test
```

There is an explanation of the contents of these directories in section 3.11. For now, only the `bin` sub–directory (the software program directory), is of concern. This sub–directory is used in section 3.7 when the `path` is modified.

It is suggested that `\ada` be used as the top–level directory. If you already have a directory named `\ada`, then choose a new name like `\mada` or `\ada.new`.

If the new compiler or optional packages are installed in the same directory as the old versions of the software, then the old versions will be wiped out. This may not be an altogether bad thing, but if you wish to preserve the old software indefinitely, just install the software using a different directory (e.g. `\ada.new`).

Be *certain* that the `path` is set appropriately and that the old `\ada\bin` directory is removed from the `path`, substituting the new `bin` sub–directory (e.g. `\ada.new\bin`).

3.6 Ensure Sufficient Disk Space Exists

Although you probably believe that you have enough disk space to install all the files, it is best to make sure. To find out how much free disk space is available, type the `dir` command. The number printed at the very end of the directory listing is the amount of free space available. For example:

```
c:
dir
```

This example shows how to determine the amount of free disk space on drive `c:`. If the number is smaller than 5000000 (that's 5,000,000 which is about 5 megabytes) then there is probably not enough space to install the software, and you must either delete some files to make room or select a different drive on which to install the software.

3.7 Make Changes to AUTOEXEC.BAT

After deciding where to install the software, the the software program directory (e.g. `c:\ada\bin`) must be added to the `path` command in the system's `autoexec.bat` file (the system startup file). The `path` command is used to tell DOS where to find various programs. DOS will be unable to find the the software programs unless `path` is set correctly.

If the `autoexec.bat` file is not present, it should be created, if only to contain a `path` command. The file is normally found in the root directory on the boot drive, e.g. `c:\autoexec.bat`. Any text editor that can be used to write Ada programs can also be used to edit the `autoexec.bat` file.

Assuming that the `autoexec.bat` file exists, something like this *example* line should appear in the `autoexec.bat` file:

```
path c:\bin;c:\mystuff
```

This is a list of directory names separated by semicolons (";"). If a `path` command does not appear, then one must be added.

The `path` command must be modified to include the `bin` sub–directory under the top–level the software installation directory. For example, if `c:\ada` is the top–level directory, then you must add `c:\ada\bin` to the `path`. The `bin` sub–directory is the software program directory. Thesoftware program directory name *always* ends with the sub–directory `bin`. The full path `c:\ada\bin` is just an *example*.

The original directories named in the `path` command should remain when the command is modified to include the `bin` sub–directory, as in this example:

```
path c:\ada\bin;c:\bin;c:\mystuff
```

This example assumes that `c:\ada` is the top–level directory where the software is installed.

Also, make sure that the last line in the `autoexec.bat` file is a *complete* line: the last line must be terminated by a newline sequence (i.e. the end of the file comes on the "line" after the last line). Most normal text editors, except for `edlin`, automatically complete the last line of a file by creating the linefeed/carriage return for that line.

If you are going to be using ACE, make sure that you add the following DOS command to your `autoexec.bat` file:

```
set acedir=c:\ada\ace
```

This example assumes that you have installed the compiler in the directory `c:\ada`.

3.8 Change the FILES Option in CONFIG.SYS

The limit on the number of open files in the system must be raised to 20. This is done by modifying (or creating) the system configuration file, `config.sys`. Any text editor that can be used to write Ada programs can also be used to edit the `config.sys` file.

If this line appears in the `config.sys` file:

```
files=20
```

then skip ahead to the next section.

If a lower value is specified for the `files` parameter, the value must be changed to 20. If this line does not appear, it must be added. If the `config.sys` file is not present (it is normally found in the root directory on the boot drive, e.g. `c:\config.sys`), it must be created, if only to contain this line.

Enter the `files` parameter (as in the above description) using any appropriate text editor.

Also make sure that the last line in the `config.sys` file is a *complete* line: the last line must be terminated by a newline sequence (i.e. the end of the file comes on the "line" after the last line). Most normal text editors, except for `edlin`, automatically complete the last line of a file by creating the linefeed/carriage return for that line.

Refer to the DOS manuals for additional information about the `config.sys` file and the `files` parameter.

This is another line that is handy to add to the `config.sys` file, but it is not required:

```
break=on
```

This makes it easier to break out of executing programs. Refer to the DOS manuals for additional information about the `break` parameter in the `config.sys` file.

The reason for raising the limit on the number of open files to 20 is that the compiler may open many files if `text_io` is used or if there are many packages in a program. A somewhat lower limit (around 15 files)

is acceptable for compiling many things, but for programs with several layers of abstraction and greater complexity, a larger number of files needs to be opened.

To make the new system configuration take effect, the system should be rebooted, as described in the next section.

3.9 Reboot the System

At this point, to effect the new settings of `files` and `path`, the system must be rebooted.

Rebooting is normally accomplished by holding down the CONTROL and ALT, keys and while holding these down, pressing the DEL key. If this fails, try turning the machine off and back on again.

3.10 Load the Files onto the Hard Disk

To load the files onto the hard disk, perform these steps:

1. Insert the first distribution diskette (marked "Disk 1") into drive `a:`.
2. Run the `install` program using a command with this form:

   `a:install` *d:\directory*

   The command line arguments to `install` are *d*, a hard disk drive letter, and *directory*, which you selected when you read section 3.5. Note that the *directory* must start with a backslash ("\").

Example:

```
a:install c:\ada
```

This causes the software to be installed on drive `c:` in the directory `\ada`.

Loading the files onto the hard disk should take several minutes. The `install` command prompts for each subsequent diskette. If there is not enough space on the destination disk to accommodate all the files, the installation will fail.

A number of directories are created by `install`. If any of the directories already exist, some harmless error messages may be printed, but installation proceeds.

If the diskette drive is not `a:`, use the DOS command `subst` to temporarily rename the drive. For example, to install the compiler from `b:`, run the following command before installing the compiler:

```
subst a: b:\
```

This renames the `b:` disk to be `a:`. After the installation is complete, the effect of the `subst` command can be undone with:

```
subst a: /d
```

3.11 Verifying the Installation

Under the top–level directory specified in the `install` command, the following sub–directories should have been created by the installation process:

`bin`	This directory contains the software programs.
`paclib`	This directory contains various low–level programs that are not directly executed by users. The directory also contains run–time libraries and configuration files for the compiler.
`test`	This directory contains test programs for verifying correct installation of the compiler (see section 3.12).

If, for example, you had specified

```
install c:\ada
```

then these directories would be present as `c:\ada\bin`, `c:\ada\paclib`, and `c:\ada\test`.

If any of these directories are missing, try the installation process from the beginning. Ensure that there is sufficient space on the hard disk drive. If installation fails again, call Meridian Support.

3.12 Testing the Software

The instructions for testing the software use the Meridian Ada command–line programs to compile and link the tests. If you wish to use ACE to run these tests, refer to the *Meridian ACE User's Guide* for information on how to compile and link programs from ACE.

3.12.1 Testing the Compiler

The distribution diskettes include several test programs that may be used to check the compiler's integrity. If any of these tests fail, consult Chapter 4 for some troubleshooting information. If all else fails, try re–installing the software. If the tests still fail, please contact Meridian's Technical Support.

Before trying these tests, make sure that the software program directory is in your path (see section 3.7). Select the drive on which the software was installed, as in this example:

```
c:
```

This example selects the `c:` drive as the current drive. If the compiler was loaded onto a different disk, then that disk should be selected (e.g. `d:` or `e:`).

After doing that, type:

```
cd \ada\test
```

This `cd` command changes the current directory to `\ada\test` on drive `c:`. If the installation directory was different than `c:\ada`, then the directory that you selected should be used. In any case, the sub–directory to use is `test`.

Next, type:

```
newlib
```

The `newlib` command creates the software library database file (`ada.lib`) in the current directory with certain default characteristics.

After creating the library database file, test the compiler by typing these commands:

```
ada simple.ada
bamp simple
```

The compiler should load and execute the first and second passes; the linker should load the executable result into `simple.exe`. If you cannot execute `ada` or `bamp`, be sure the software program directory is in your execution path (see section 3.7). Assuming the compilation completed, type:

```
simple
```

The output should look like this:

```
1 2 3 4 5 6 7 8 9 10
```

If this was successful, try a slightly more difficult program:

```
ada sieve.ada
bamp sieve
sieve
```

`Sieve` computes and prints the first 1899 primes. The last prime should be 16381.

If these programs work correctly, the compiler was probably loaded correctly and is fully operational.

There are some additional test source programs provided; try compiling, linking, and running them. Some of the programs in the `test` directory have components in separate files and must be compiled separately. The `read.me` file tells you the order in which to compile the test programs.

Do not use `bamp` on every compilation unit. Some of the compilation units are just *packages*, not main programs. Refer to the *Meridian Ada Compiler User's Guide* for specific information on how to use `bamp`.

3.12.2 Testing the Debugger

The following test programs can be used to check the Debugger feature of the compiler:

- `dbtest.ada`
- `dbtask.ada`

To test the Debugger, the software library must be created using the normal `newlib` command. To compile a test program for debugging, the `-fD` flag must be specified.

```
ada -fD dbtest.ada
```

This flag (lowercase `f` and uppercase `D`) instructs the compiler to insert debugging code into the object code. It also makes a note in the program library that the unit has been compiled for debugging. This can be confirmed by using the `lslib` command:

```
lslib -l dbtest
```

The result should include the line "`Uses debugger.`"

To create an executable file, the `bamp` command is run as usual.

```
bamp dbtest
```

The `bamp` command automatically links the Debugger with the program if the program was compiled with the `ada -fD` option for debugging.

To execute the Debugger, simply run the debugged program:

```
dbtest
```

The program begins to run and encounters the default breakpoint at the start of the program. Some text identifying the Debugger is printed. The prompt is the right angle bracket ("`>`") character. At this point, try typing the `help` command. The available commands are listed. Next, try the `ss` (single step) command several times. Successive lines of source should be printed as they are executed. Display the main subprogram with the `print` command:

```
> print(dbtest)
```

The values of the objects declared in the main subprogram may be examined simply by typing their names to the Debugger:

```
> x
37
> s
"abc"
```

If the commands have worked as indicated, the Debugger should be installed properly. Try compiling and running some of the other test programs to familiarize yourself with the Debugger. These test programs are located in the `test` directory.

3.13 Installation and Test Summary

This section summarizes the commands required to install and test the compiler; *it is an example only* – the actual commands may differ slightly.

```
REM -- It is assumed that FILES=20 in CONFIG.SYS.
REM -- It is also assumed that PATH contains
REM -- the appropriate directory, as in this command:
path c:\ada\bin;c:\bin
REM -- You should also define ACEDIR, as in this command:
set acedir=c:\ada\ace
REM -- Perform the installation procedure. This is an example.
a:install c:\ada
REM -- Select the hard disk.
c:
REM -- Select the test directory.
cd \ada\test
REM -- Create the library.
newlib
REM -- Compile, link, and run the sample programs.
ada simple.ada
bamp simple
simple
ada sieve.ada
bamp sieve
sieve
```

Chapter 4 Troubleshooting the Installation

This chapter provides help for some problems which you may encounter during installation. Your first resort should be to read any *Release Notes* that accompany the compiler. These *Release Notes* sometimes contain some last minute documentation updates. An additional troubleshooting section is contained in the *Meridian Ada Compiler User's Guide*. That trouble shooting section deals primarily with errors and problems that might occur during the compilation process.

4.1 Installation Fails Mysteriously

Possible solutions are:

- Reboot the system, then try the installation procedure again. Rebooting ensures that the system starts with a clean slate.
- Make sure that no resident programs are running (for example, print spooler programs, window managers, etc.). These may include network drivers and unusually large device drivers that consume portions of base memory (the main 640K area). Use the `ada -fv` option to determine memory consumption during compilation. If, for example, you are starting out with only 60K of memory available for symbol table storage (the amount available appears in [square brackets]). you will run into trouble if your compilation unit uses `text_io` or other large packages.

If things "freeze up" (i.e. your computer stops doing things and Control–C (^C) doesn't recover) while `install` says it is running the `checkarg` program, then the problem is most likely related to the math co–processor hardware set–up on your system. On old IBM PCs or IBM PC/XT's, this means that a DIP switch is incorrectly indicating the presence of a math co–processor. On newer systems, a software configuration program must be run.

4.1.1 Bad Command or Filename

You receive the following error message:

```
bad command or filename
```

Possible solutions:

- Make sure that `path` is set correctly; type the command `path` to find out. Rebooting the system (see section 3.9) may cause whatever automatic startup procedures are already in place to set the `path` correctly. Also ensure that you have followed the procedures in section 3.7.
- Make sure that the command was typed correctly.
- Make sure that the command program (`xxx.exe`, `xxx.com`, or `xxx.bat` for a command named `xxx`) is actually installed in a directory mentioned in the `path`.

4.1.2 Missing Distribution Files

You receive the following error message:

`missing distribution files`

Possible solutions:

- Make sure that the installation completed and that all directories described in section 3.11 are present.
- Read the Release Notes that accompany the software to check for any additional information.
- Make sure that you actually have enough disk space. See section 3.6.

4.1.3 Missing Error Message File

You receive the following error message:

`cannot find error file err.msg`

Possible solutions:

- Make sure that **`files=20`** in the **`config.sys`** file (see section 3.8). Also make sure that the last line in the **`config.sys`** file is a *complete* line: the last line must be terminated by a newline sequence (i.e. the end of the file comes on the "line" after the last line). Most normal text editors, except for **`edlin`**, automatically complete the last line of a file with a carriage return/linefeed sequence.
- Make sure that the **`err.msg`** file in the **`paclib`** sub–directory (e.g. **`c:\ada\paclib`**) is actually present; if it is not, try re–installing the software from scratch.

4.1.4 Missing Batch File

You receive the following error message:

`batch file missing`

Possible solution:

- This message may appear during installation if you have switched disks too soon. Re–install the software this time making sure that you only change the diskette when explicitly prompted to do so. Note that the very first pause in the installation process is made to verify that the execution path has been configured appropriately, *not* to request that you change disks.
- Make sure that the **`path`** is set correctly; type the command **`path`** to find out. Also ensure that you have followed the procedures in section 3.7.

Index

P

R

S

T

U

V

W